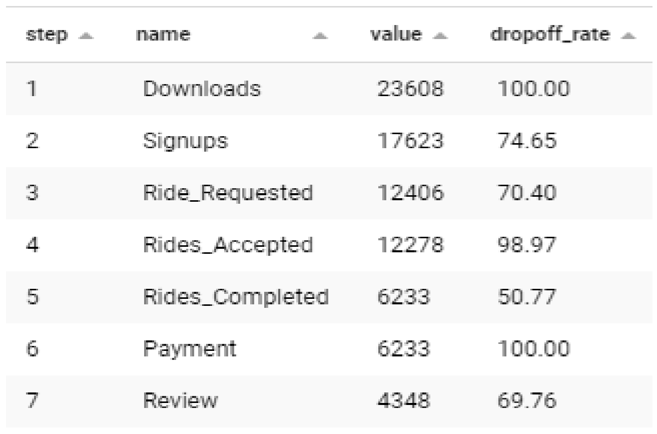
Metrocar Funnel Analysis SQL Queries

QUESTION:1 What steps of the funnel should we research and improve? Are there any specific drop-off points preventing users from completing their first ride?

User level:

```
-- Common Table Expression (CTE) new_table to calculate counts for each step
WITH new_table AS (
  SELECT
    1 AS step,
    'Downloads' AS name,
    COUNT(DISTINCT app_download_key) AS value
  FROM
    app_downloads
  UNION
  SELECT
    2 AS step,
    'Signups' AS name,
    COUNT(DISTINCT user_id) AS value
  FROM
    signups
  UNION
  SELECT
    3 AS step,
    'Ride_Requested' AS name,
    COUNT(DISTINCT user_id) AS value
  FROM
    ride_requests
  UNION
  SELECT
    4 AS step,
    'Rides_Accepted' AS name,
    COUNT(DISTINCT user_id) AS value
  FROM
    ride_requests
  WHERE
    accept_ts IS NOT NULL
  UNION
  SELECT
    5 AS step,
    'Rides_Completed' AS name,
    COUNT(DISTINCT user_id) AS value
  FROM
    ride_requests
  WHERE
    pickup_ts IS NOT NULL
    AND dropoff_ts IS NOT NULL
  UNION
  SELECT
    6 AS step,
    'Payment' AS name,
    COUNT(DISTINCT r.user_id) AS value
  FROM
    ride_requests AS r
    INNER JOIN transactions AS t ON r.ride_id = t.ride_id
  WHERE
    charge_status = 'Approved'
  UNION
  SELECT
    7 AS step,
    'Review' AS name,
    COUNT(DISTINCT user_id) AS value
  FROM
    reviews
  ORDER BY
    step
)

-- Main query to calculate the dropoff rate
SELECT
  *,
  COALESCE(
    ROUND(
      (value::decimal / LAG(value, 1) OVER(ORDER BY step) * 100),
      2
    ),
    100.00
  ) AS dropoff_rate
FROM
  new_table;
```

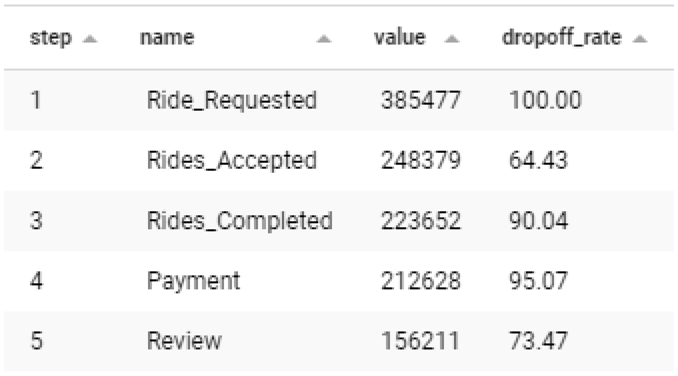


step	name	value	dropoff_rate
1	Downloads	23608	100.00
2	Signups	17623	74.65
3	Ride_Requested	12406	70.40
4	Rides_Accepted	12278	98.97
5	Rides_Completed	6233	50.77
6	Payment	6233	100.00
7	Review	4348	69.76

Ride level:

```
-- Common Table Expression (CTE) new_table to calculate counts for each step
WITH new_table AS (
 SELECT
  1 AS step,
  'Ride_Requested' AS name,
  COUNT(DISTINCT ride_id) AS value
 FROM
  ride_requests
 UNION
 SELECT
  2 AS step,
  'Rides_Accepted' AS name,
  COUNT(DISTINCT ride_id) AS value
 FROM
  ride_requests
 WHERE
  accept_ts IS NOT NULL
 UNION
 SELECT
  3 AS step,
  'Rides_Completed' AS name,
  COUNT(DISTINCT ride_id) AS value
 FROM
  ride_requests
 WHERE
  pickup_ts IS NOT NULL
  AND dropoff_ts IS NOT NULL
 UNION
```

```
  SELECT
    4 AS step,
    'Payment' AS name,
    COUNT(DISTINCT r.ride_id) AS value
  FROM
    ride_requests AS r
    INNER JOIN transactions AS t ON r.ride_id = t.ride_id
  WHERE
    charge_status = 'Approved'
  UNION
  SELECT
    5 AS step,
    'Review' AS name,
    COUNT(DISTINCT ride_id) AS value
  FROM
    reviews
  ORDER BY
    step
)
-- Main query to calculate the dropoff rate
SELECT
  *,
  COALESCE(
    ROUND(
      (value::decimal / LAG(value, 1) OVER(ORDER BY step) * 100),
      2
    ),
    100.00
  ) AS dropoff_rate
FROM
  new_table
```



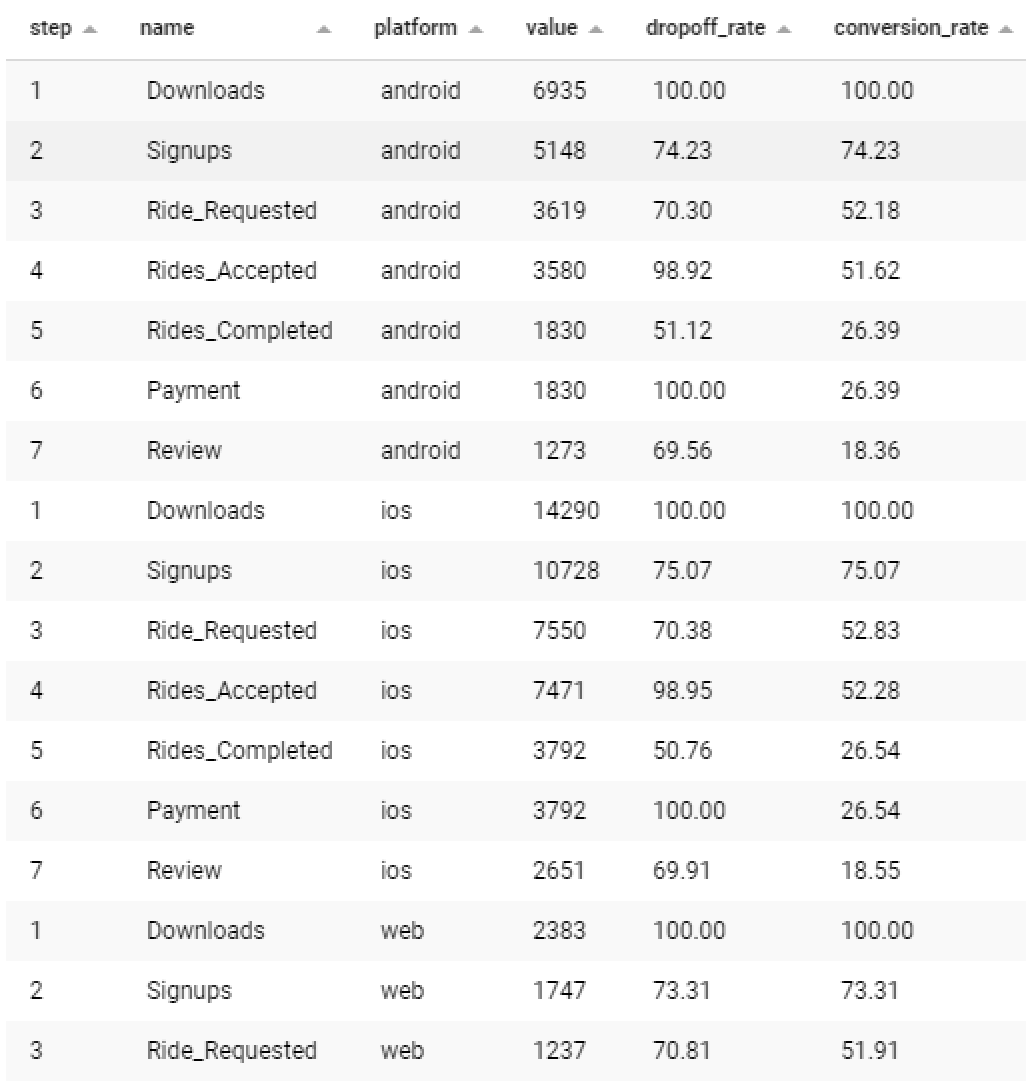
step	name	value	dropoff_rate
1	Ride_Requested	385477	100.00
2	Rides_Accepted	248379	64.43
3	Rides_Completed	223652	90.04
4	Payment	212628	95.07
5	Review	156211	73.47

QUESTION 2: Metrocar currently supports 3 different platforms: ios, android, and web. To recommend where to focus our marketing budget for the upcoming year, what insights can we make based on the platform?

User Level:

```
-- Common Table Expression (CTE) to compute the aggregated data
WITH new_table AS (
   SELECT a.platform AS platform,
          COUNT(DISTINCT a.app_download_key) AS download_users,
          COUNT(DISTINCT s.user_id) AS signups_users,
          COUNT(DISTINCT r.user_id) AS ride_requested_users,
          COUNT(DISTINCT CASE WHEN r.accept_ts IS NOT NULL THEN r.user_id END) AS
ride_accepted_users,
          COUNT(DISTINCT CASE WHEN r.pickup_ts IS NOT NULL AND r.dropoff_ts IS NOT
NULL THEN r.user_id END) AS ride_completed_users,
          COUNT(DISTINCT CASE WHEN tr.charge_status = 'Approved' THEN r.user_id END) AS
payment_users,
          COUNT(DISTINCT rw.user_id) AS review_users
   FROM app_downloads AS a
   LEFT JOIN signups AS s ON a.app_download_key = s.session_id
   LEFT JOIN ride_requests AS r ON s.user_id = r.user_id
   LEFT JOIN transactions AS tr ON r.ride_id = tr.ride_id
   LEFT JOIN reviews AS rw ON r.user_id = rw.user_id
   GROUP BY a.platform
),

-- Common Table Expression (CTE) to union the data and define steps
union_table AS (
   SELECT 1 AS step,
          'Downloads' AS name,
          platform,
          download_users AS value
   FROM new_table
   UNION
   SELECT 2 AS step,
          'Signups' AS name,
          platform,
          signups_users AS value
   FROM new_table
   UNION
```

```
    SELECT 3 AS step,
           'Ride_Requested' AS name,
           platform,
           ride_requested_users AS value
    FROM new_table
    UNION
    SELECT 4 AS step,
           'Rides_Accepted' AS name,
           platform,
           ride_accepted_users AS value
    FROM new_table
    UNION
    SELECT 5 AS step,
           'Rides_Completed' AS name,
           platform,
           ride_completed_users AS value
    FROM new_table
    UNION
    SELECT 6 AS step,
           'Payment' AS name,
           platform,
           payment_users AS value
    FROM new_table
    UNION
    SELECT 7 AS step,
           'Review' AS name,
           platform,
           review_users AS value
    FROM new_table
    ORDER BY platform, step
)

-- Final query with necessary calculations
SELECT *,
       COALESCE(ROUND((value::decimal / LAG(value, 1) OVER(PARTITION BY platform
ORDER BY step) * 100), 2), 100.00) AS dropoff_rate,
       ROUND(((value)::decimal / FIRST_VALUE(value) OVER(PARTITION BY platform ORDER
BY step) * 100), 2) AS conversion_rate
FROM union_table;
```

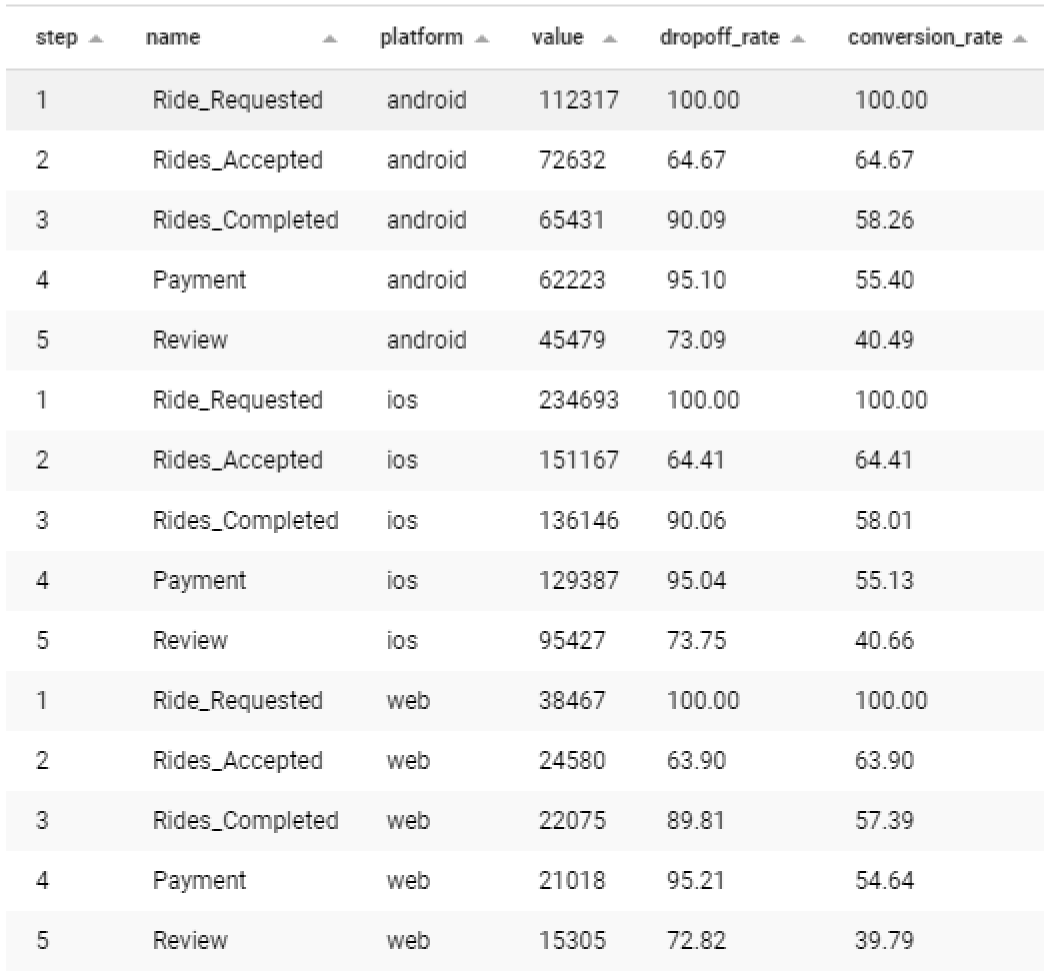


step	name	platform	value	dropoff_rate	conversion_rate
1	Downloads	android	6935	100.00	100.00
2	Signups	android	5148	74.23	74.23
3	Ride_Requested	android	3619	70.30	52.18
4	Rides_Accepted	android	3580	98.92	51.62
5	Rides_Completed	android	1830	51.12	26.39
6	Payment	android	1830	100.00	26.39
7	Review	android	1273	69.56	18.36
1	Downloads	ios	14290	100.00	100.00
2	Signups	ios	10728	75.07	75.07
3	Ride_Requested	ios	7550	70.38	52.83
4	Rides_Accepted	ios	7471	98.95	52.28
5	Rides_Completed	ios	3792	50.76	26.54
6	Payment	ios	3792	100.00	26.54
7	Review	ios	2651	69.91	18.55
1	Downloads	web	2383	100.00	100.00
2	Signups	web	1747	73.31	73.31
3	Ride_Requested	web	1237	70.81	51.91

Ride Level:

```
-- Common Table Expression (CTE) to compute the aggregated data
WITH new_table AS (
    SELECT a.platform AS platform,
           COUNT(DISTINCT r.ride_id) AS total_ride_requested,
           COUNT(DISTINCT CASE WHEN r.accept_ts IS NOT NULL THEN r.ride_id END) AS
total_ride_accepted,
           COUNT(DISTINCT CASE WHEN r.pickup_ts IS NOT NULL AND r.dropoff_ts IS NOT
NULL THEN r.ride_id END) AS total_ride_completed,
           COUNT(DISTINCT CASE WHEN tr.charge_status = 'Approved' THEN r.ride_id END) AS
ride_payment,
        COUNT(DISTINCT rw.ride_id) AS ride_review
    FROM app_downloads AS a
    LEFT JOIN signups AS s ON a.app_download_key = s.session_id
    LEFT JOIN ride_requests AS r ON s.user_id = r.user_id
    LEFT JOIN transactions AS tr ON r.ride_id = tr.ride_id
    LEFT JOIN reviews AS rw ON r.user_id = rw.user_id
    GROUP BY a.platform
),

-- Common Table Expression (CTE) to union the data and define steps
union_table AS (
    SELECT 1 AS step,
        'Ride_Requested' AS name,
        platform,
        total_ride_requested AS value
    FROM new_table
    UNION
    SELECT 2 AS step,
        'Rides_Accepted' AS name,
        platform,
        total_ride_accepted AS value
    FROM new_table
    UNION
    SELECT 3 AS step,
        'Rides_Completed' AS name,
        platform,
        total_ride_completed AS value
    FROM new_table
    UNION
    SELECT 4 AS step,
        'Payment' AS name,
        platform,
        ride_payment AS value
    FROM new_table
    UNION
    SELECT 5 AS step,
        'Review' AS name,
        platform,
        ride_review AS value
    FROM new_table
    ORDER BY platform, step
)

-- Final query with necessary calculations
```

```
SELECT *,
    COALESCE(ROUND((value::decimal / NULLIF(LAG(value) OVER(PARTITION BY platform
ORDER BY step), 0) * 100), 2), 100.00) AS dropoff_rate,
    ROUND(((value)::decimal / FIRST_VALUE(value) OVER(PARTITION BY platform ORDER
BY step) * 100), 2) AS conversion_rate
FROM union_table;
```

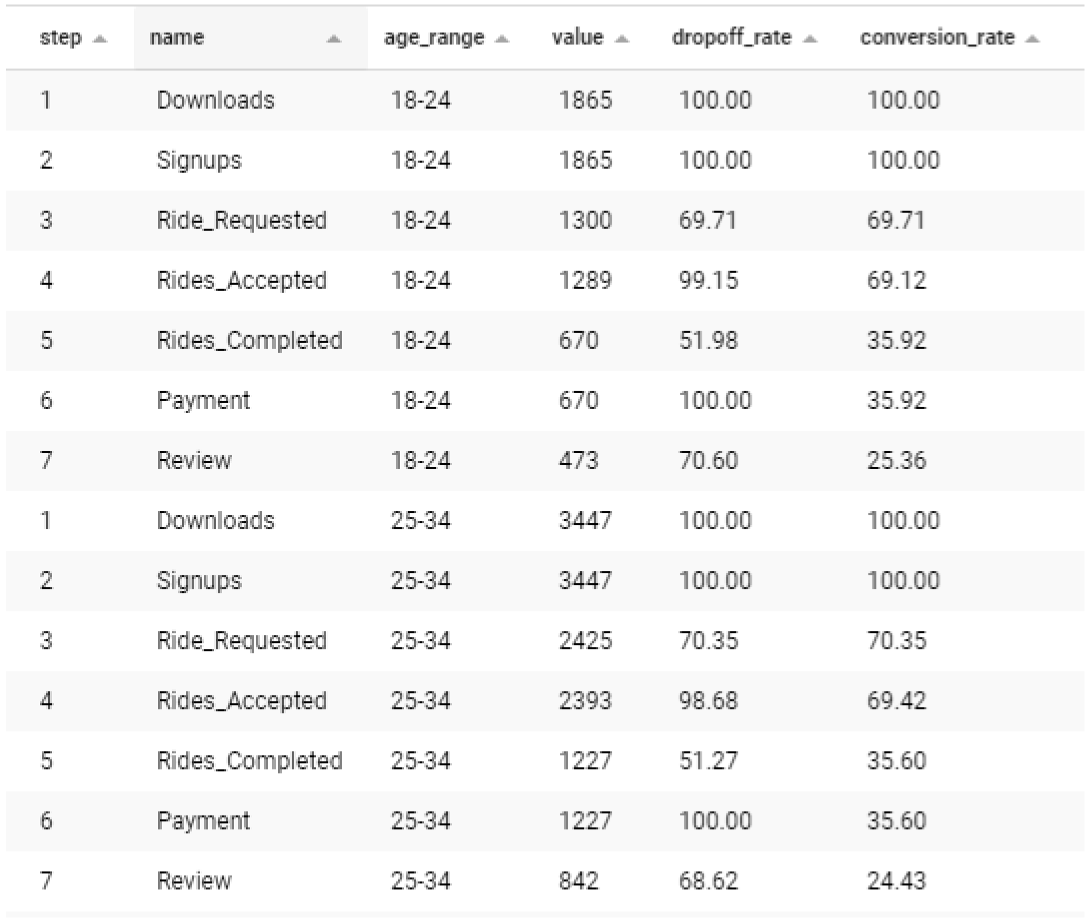


step	name	platform	value	dropoff_rate	conversion_rate
1	Ride_Requested	android	112317	100.00	100.00
2	Rides_Accepted	android	72632	64.67	64.67
3	Rides_Completed	android	65431	90.09	58.26
4	Payment	android	62223	95.10	55.40
5	Review	android	45479	73.09	40.49
1	Ride_Requested	ios	234693	100.00	100.00
2	Rides_Accepted	ios	151167	64.41	64.41
3	Rides_Completed	ios	136146	90.06	58.01
4	Payment	ios	129387	95.04	55.13
5	Review	ios	95427	73.75	40.66
1	Ride_Requested	web	38467	100.00	100.00
2	Rides_Accepted	web	24580	63.90	63.90
3	Rides_Completed	web	22075	89.81	57.39
4	Payment	web	21018	95.21	54.64
5	Review	web	15305	72.82	39.79

QUESTION 3: What age groups perform best at each stage of our funnel? Which age group(s) likely contain our target customers?

User Level:

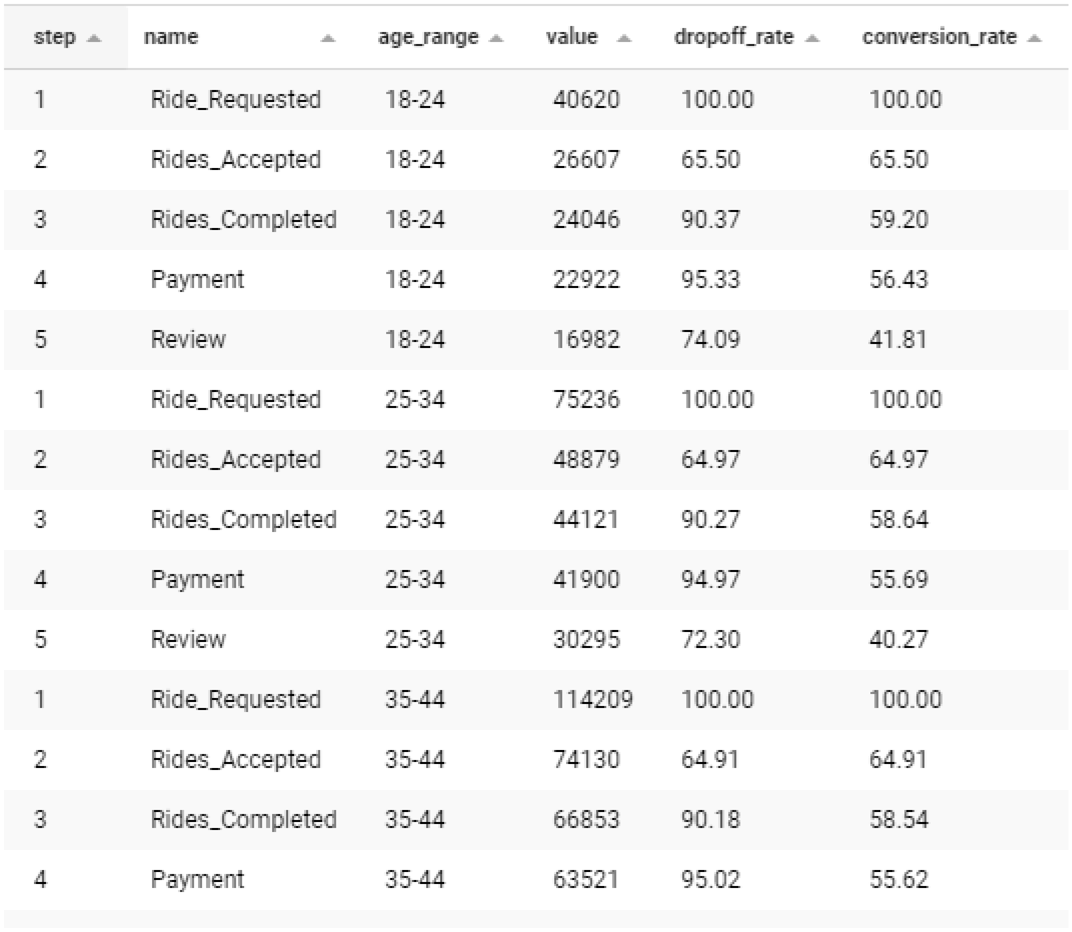
```
-- Define CTE new_table to compute counts for each age range and each stage
WITH new_table AS (
   SELECT s.age_range AS age_range,
          COUNT(DISTINCT a.app_download_key) AS download_users,
          COUNT(DISTINCT s.user_id) AS signups_users,
          COUNT(DISTINCT r.user_id) AS ride_requested_users,
          COUNT(DISTINCT CASE WHEN r.accept_ts IS NOT NULL THEN r.user_id END) AS
ride_accepted_users,
          COUNT(DISTINCT CASE WHEN r.pickup_ts IS NOT NULL AND r.dropoff_ts IS NOT
NULL THEN r.user_id END) AS ride_completed_users,
          COUNT(DISTINCT CASE WHEN tr.charge_status = 'Approved' THEN r.user_id END) AS
payment_users,
          COUNT(DISTINCT rw.user_id) AS review_users
   FROM app_downloads AS a
   LEFT JOIN signups AS s ON a.app_download_key = s.session_id
   LEFT JOIN ride_requests AS r ON s.user_id = r.user_id
   LEFT JOIN transactions AS tr ON r.ride_id = tr.ride_id
   LEFT JOIN reviews AS rw ON r.user_id = rw.user_id
   GROUP BY s.age_range
),
-- Define CTE union_table to merge results from new_table for each stage
union_table AS (
   SELECT 1 AS step,
          'Downloads' AS name,
          age_range,
          download_users AS value
   FROM new_table
   UNION
   SELECT 2 AS step,
          'Signups' AS name,
          age_range,
          signups_users AS value
   FROM new_table
   UNION
   SELECT 3 AS step,
          'Ride_Requested' AS name,
        age_range,
        ride_requested_users AS value
    FROM new_table
    UNION
    SELECT 4 AS step,
        'Rides_Accepted' AS name,
        age_range,
        ride_accepted_users AS value
    FROM new_table
    UNION
    SELECT 5 AS step,
        'Rides_Completed' AS name,
        age_range,
        ride_completed_users AS value
    FROM new_table
    UNION
    SELECT 6 AS step,
        'Payment' AS name,
        age_range,
        payment_users AS value
    FROM new_table
    UNION
    SELECT 7 AS step,
        'Review' AS name,
        age_range,
        review_users AS value
    FROM new_table
    ORDER BY age_range, step
)

-- Final query to calculate conversion rate and drop-off rate
SELECT *,
    -- Calculate dropoff_rate and handle division by zero
    COALESCE(
        ROUND(
            (value::decimal /
             CASE
                WHEN LAG(value, 1) OVER (PARTITION BY age_range ORDER BY step) = 0 THEN
1
                ELSE LAG(value, 1) OVER (PARTITION BY age_range ORDER BY step)
             END
             * 100
            ),
            2
        ),
        100.00
    ) AS dropoff_rate,
    -- Calculate conversion_rate rate
    ROUND(
        ((value)::decimal / FIRST_VALUE(value) OVER (PARTITION BY age_range ORDER BY
step) * 100),
        2
    ) AS conversion_rate
FROM union_table;
```



step	name	age_range	value	dropoff_rate	conversion_rate
1	Downloads	18-24	1865	100.00	100.00
2	Signups	18-24	1865	100.00	100.00
3	Ride_Requested	18-24	1300	69.71	69.71
4	Rides_Accepted	18-24	1289	99.15	69.12
5	Rides_Completed	18-24	670	51.98	35.92
6	Payment	18-24	670	100.00	35.92
7	Review	18-24	473	70.60	25.36
1	Downloads	25-34	3447	100.00	100.00
2	Signups	25-34	3447	100.00	100.00
3	Ride_Requested	25-34	2425	70.35	70.35
4	Rides_Accepted	25-34	2393	98.68	69.42
5	Rides_Completed	25-34	1227	51.27	35.60
6	Payment	25-34	1227	100.00	35.60
7	Review	25-34	842	68.62	24.43

Ride Level:

```
-- Define CTE new_table to compute counts for each age range and each stage
WITH new_table AS (
    SELECT s.age_range AS age_range,
            COUNT(DISTINCT r.ride_id) AS total_ride_requested,
            COUNT(DISTINCT CASE WHEN r.accept_ts IS NOT NULL THEN r.ride_id END) AS
total_ride_accepted,
            COUNT(DISTINCT CASE WHEN r.pickup_ts IS NOT NULL AND r.dropoff_ts IS NOT
NULL THEN r.ride_id END) AS total_ride_completed,
            COUNT(DISTINCT CASE WHEN tr.charge_status = 'Approved' THEN r.ride_id END) AS
ride_payment,
            COUNT(DISTINCT rw.ride_id) AS ride_review
    FROM app_downloads AS a
    LEFT JOIN signups AS s ON a.app_download_key = s.session_id
    LEFT JOIN ride_requests AS r ON s.user_id = r.user_id
    LEFT JOIN transactions AS tr ON r.ride_id = tr.ride_id
    LEFT JOIN reviews AS rw ON r.user_id = rw.user_id
    GROUP BY s.age_range
),
-- Define CTE union_table to merge results from new_table for each stage
union_table AS (
    SELECT 1 AS step,
           'Ride_Requested' AS name,
           age_range,
           total_ride_requested AS value
    FROM new_table
    UNION
    SELECT 2 AS step,
           'Rides_Accepted' AS name,
           age_range,
           total_ride_accepted AS value
    FROM new_table
    UNION
    SELECT 3 AS step,
           'Rides_Completed' AS name,
           age_range,
           total_ride_completed AS value
    FROM new_table
    UNION
    SELECT 4 AS step,
           'Payment' AS name,
           age_range,
           ride_payment AS value
    FROM new_table
    UNION
    SELECT 5 AS step,
           'Review' AS name,
           age_range,
        ride_review AS value
  FROM new_table
  ORDER BY age_range, step
)

SELECT *,
  -- Calculate drop rate and handle division by zero
  COALESCE(
    ROUND(
      (value::decimal /
       CASE
         WHEN LAG(value, 1) OVER (PARTITION BY age_range ORDER BY step) = 0 THEN
1
         ELSE LAG(value, 1) OVER (PARTITION BY age_range ORDER BY step)
       END
       * 100
      ),
      2
    ),
    100.00
  ) AS dropoff_rate,
  -- Calculate conversion_rate rate and handle division by zero
  ROUND(
    ((value)::decimal /
     CASE
       WHEN FIRST_VALUE(value) OVER (PARTITION BY age_range ORDER BY step) = 0
THEN 1
       ELSE FIRST_VALUE(value) OVER (PARTITION BY age_range ORDER BY step)
     END
     * 100
    ),
    2
  ) AS conversion_rate
FROM union_table;
```

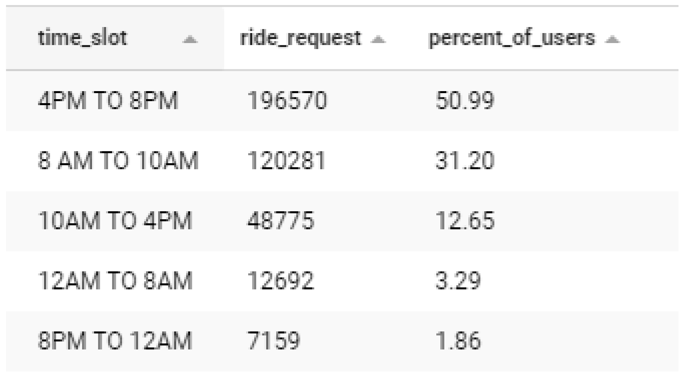


step	name	age_range	value	dropoff_rate	conversion_rate
1	Ride_Requested	18-24	40620	100.00	100.00
2	Rides_Accepted	18-24	26607	65.50	65.50
3	Rides_Completed	18-24	24046	90.37	59.20
4	Payment	18-24	22922	95.33	56.43
5	Review	18-24	16982	74.09	41.81
1	Ride_Requested	25-34	75236	100.00	100.00
2	Rides_Accepted	25-34	48879	64.97	64.97
3	Rides_Completed	25-34	44121	90.27	58.64
4	Payment	25-34	41900	94.97	55.69
5	Review	25-34	30295	72.30	40.27
1	Ride_Requested	35-44	114209	100.00	100.00
2	Rides_Accepted	35-44	74130	64.91	64.91
3	Rides_Completed	35-44	66853	90.18	58.54
4	Payment	35-44	63521	95.02	55.62

QUESTION 4: **Surge pricing is the practice of increasing the price of goods or services when there is the greatest demand for them. If we want to adopt a price-surging strategy, what does the distribution of ride requests look like throughout the day?**

Peak Hour Distribution:

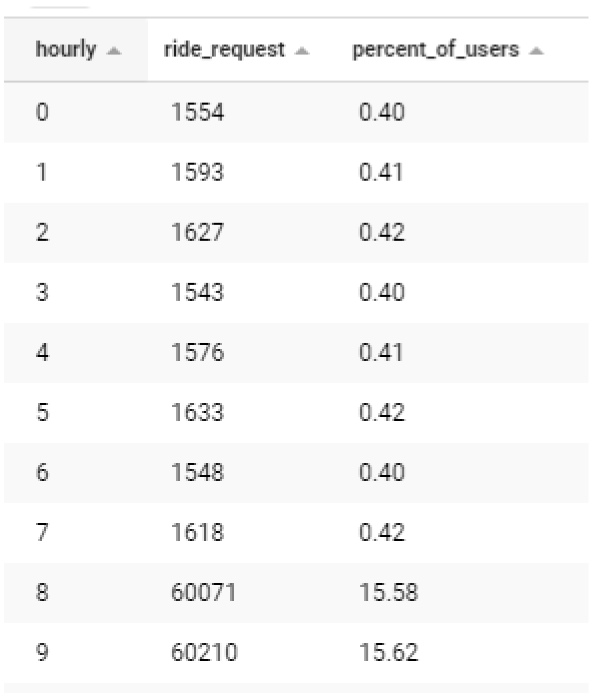
```
WITH RideRequests AS(
SELECT
   -- Categorize time slots based on the hour extracted from request_ts
   CASE
      WHEN EXTRACT(HOUR FROM request_ts) >= 8 AND EXTRACT(HOUR FROM
request_ts) < 10 THEN '8 AM TO 10AM'
      WHEN EXTRACT(HOUR FROM request_ts) >= 10 AND EXTRACT(HOUR FROM
request_ts) < 16 THEN '10AM TO 4PM'
      WHEN EXTRACT(HOUR FROM request_ts) >= 16 AND EXTRACT(HOUR FROM
request_ts) < 20 THEN '4PM TO 8PM'
      WHEN EXTRACT(HOUR FROM request_ts) >= 20 AND EXTRACT(HOUR FROM
request_ts) <= 24 THEN '8PM TO 12AM'
      WHEN EXTRACT(HOUR FROM request_ts) >= 0 AND EXTRACT(HOUR FROM
request_ts) < 8 THEN '12AM TO 8AM'
      ELSE 'time'
   END AS time_slot,
   COUNT(ride_id) AS ride_request
FROM ride_requests
-- Group the results by the time slots
GROUP BY time_slot
-- Order the results by the count of ride requests in descending order
ORDER BY ride_request DESC
)

-- Final query to display time slots, ride requests, and percentage of users in each time slot
SELECT
   time_slot,
   ride_request,
   ROUND((ride_request * 100.0) / SUM(ride_request) OVER (), 2) AS percent_of_users
FROM RideRequests;
```



time_slot	ride_request	percent_of_users
4PM TO 8PM	196570	50.99
8 AM TO 10AM	120281	31.20
10AM TO 4PM	48775	12.65
12AM TO 8AM	12692	3.29
8PM TO 12AM	7159	1.86

Hourly Distribution:

```
WITH RideRequests AS(
-- Extract the hour from the request timestamp
SELECT EXTRACT(HOUR FROM request_ts) AS hourly,
      -- Count the number of ride requests
      COUNT(ride_id) AS ride_request
FROM ride_requests
GROUP BY EXTRACT(HOUR FROM request_ts)
ORDER BY hourly
)

-- Final query to display hour, ride requests, and percentage of users in each hour
SELECT
   hourly,
   ride_request,
   ROUND((ride_request * 100.0) / SUM(ride_request) OVER (), 2) AS percent_of_users
FROM RideRequests;
```

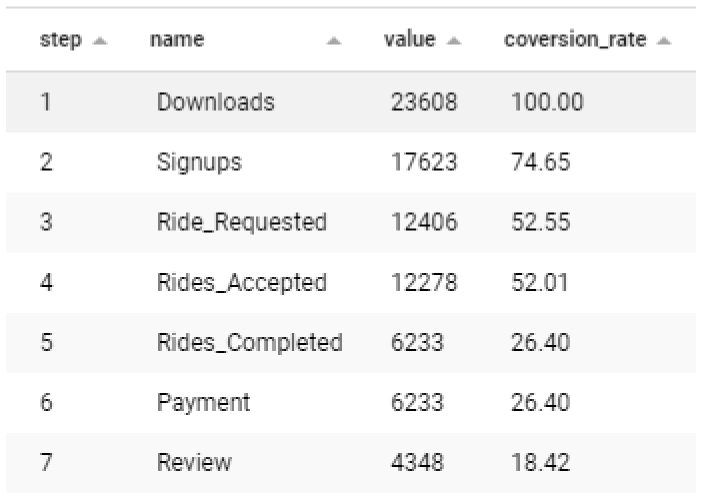


hourly	ride_request	percent_of_users
0	1554	0.40
1	1593	0.41
2	1627	0.42
3	1543	0.40
4	1576	0.41
5	1633	0.42
6	1548	0.40
7	1618	0.42
8	60071	15.58
9	60210	15.62

QUESTION 5: What part of our funnel has the lowest conversion rate? What can we do to improve this part of the funnel?

User Level:

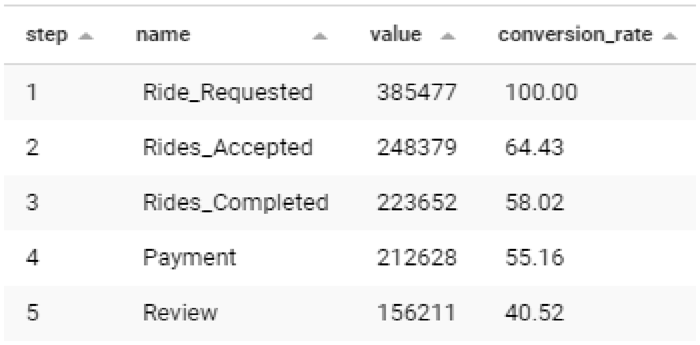
```
-- Calculate aggregated data for each step
WITH new_table AS (
  SELECT 1 AS step,
         'Downloads' AS name,
         COUNT(DISTINCT app_download_key) AS value
  FROM app_downloads
  UNION
  SELECT 2 AS step,
         'Signups' AS name,
         COUNT(DISTINCT user_id) AS value
  FROM signups
  UNION
  SELECT 3 AS step,
         'Ride_Requested' AS name,
         COUNT(DISTINCT user_id) AS value
  FROM ride_requests
  UNION
  SELECT 4 AS step,
         'Rides_Accepted' AS name,
         COUNT(DISTINCT user_id) AS value
  FROM ride_requests
  WHERE accept_ts IS NOT NULL
  UNION
  SELECT 5 AS step,
         'Rides_Completed' AS name,
         COUNT(DISTINCT user_id) AS value
  FROM ride_requests
  WHERE pickup_ts IS NOT NULL AND dropoff_ts IS NOT NULL
  UNION
  SELECT 6 AS step,
         'Payment' AS name,
         COUNT(DISTINCT r.user_id) AS value
  FROM ride_requests AS r
  INNER JOIN transactions AS t ON r.ride_id = t.ride_id
  WHERE charge_status = 'Approved'
  UNION
  SELECT 7 AS step,
        'Review' AS name,
        COUNT(DISTINCT user_id) AS value
  FROM reviews
  ORDER BY step
)

-- Calculate the coversion_rate rate
SELECT
  *,
  ROUND(((value)::decimal / FIRST_VALUE(value) OVER(ORDER BY step) * 100), 2) AS
coversion_rate
FROM
  new_table;
```



step	name	value	coversion_rate
1	Downloads	23608	100.00
2	Signups	17623	74.65
3	Ride_Requested	12406	52.55
4	Rides_Accepted	12278	52.01
5	Rides_Completed	6233	26.40
6	Payment	6233	26.40
7	Review	4348	18.42

Ride level:

```
-- Calculate aggregated data for each step
WITH new_table AS (
  SELECT 1 AS step,
        'Ride_Requested' AS name,
        COUNT(DISTINCT ride_id) AS value
  FROM ride_requests
  UNION
  SELECT 2 AS step,
        'Rides_Accepted' AS name,
        COUNT(DISTINCT ride_id) AS value
  FROM ride_requests
  WHERE accept_ts IS NOT NULL
  UNION
  SELECT 3 AS step,
        'Rides_Completed' AS name,
        COUNT(DISTINCT ride_id) AS value
  FROM ride_requests
  WHERE pickup_ts IS NOT NULL AND dropoff_ts IS NOT NULL
  UNION
  SELECT 4 AS step,
        'Payment' AS name,
        COUNT(DISTINCT r.ride_id) AS value
  FROM ride_requests AS r
  INNER JOIN transactions AS t ON r.ride_id = t.ride_id
  WHERE charge_status = 'Approved'
  UNION
  SELECT 5 AS step,
        'Review' AS name,
        COUNT(DISTINCT ride_id) AS value
  FROM reviews
  ORDER BY step
)
-- Calculate the conversion rate
SELECT
  *,
  ROUND(((value)::decimal / FIRST_VALUE(value) OVER(ORDER BY step) * 100), 2) AS
conversion_rate
FROM
  new_table;
```



step	name	value	conversion_rate
1	Ride_Requested	385477	100.00
2	Rides_Accepted	248379	64.43
3	Rides_Completed	223652	58.02
4	Payment	212628	55.16
5	Review	156211	40.52